## Remarks

Applicant respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. Claims 7, 14, 15, 21, 24-37 and 41-55 are pending in the application. Claims 7, 14, 15, 21 and 24-37 are rejected. Claims 1-6, 8-13, 16-20, 22, 23 and 38-40 are canceled without prejudice. Claims 7, 24, 30, 34, 47, 50 and 53 are independent. Claims 41-55 are new.

### *Allowability of Claims Under 35 U.S.C. § 101*

Claims 6 and 38-40 were rejected under 35 U.S.C. § 101 but have been canceled without prejudice. The claims in their present form are allowable under 35 U.S.C. § 101.

### *Allowability of Claims Under 35 U.S.C. 102*

The Action rejects claim 7 under 35 U.S.C. § 102(b) as allegedly being anticipated by Bertrand Meyer, "Eiffel: Programming for Reusability and Extendibility," *ACM SIGPLAN Notices*, vol. 22, no. 2, pp. 85-94, February 1987 (hereinafter, "Meyer").

The Action also rejects claims 7, 14, 15, 21 and 24-37 under 35 U.S.C. § 102(a) as allegedly being anticipated by Evans et al., "Splint Manual, Version 3.1.1-1," (hereinafter, "Splint manual"). The Splint manual is dated "5 June 2003." Applicants do not admit that this Splint manual is prior art to the present application and reserve the right to provide evidence of prior conception.

For a rejection under 35 U.S.C. § 102 to be proper, the applied art must show each and every element as set forth in a claim. (*See* MPEP § 2131.) Applicants respectfully submit that the claims in their present form are allowable over the applied art because it does not teach or suggest all the claim limitations of claims 7, 14, 15, 21 and 24-37.

Applicants respectfully traverse these rejections and submit that the claims in their present form are allowable over the applied art.

*Independent Claim 7*

As amended, independent claim 7 recites in part:

> the one or more annotations comprise at least one annotation on a first pointer to a buffer, wherein the at least one annotation comprises a property that indicates a characteristic of the buffer, wherein the property that indicates the characteristic of the buffer takes a size argument, and wherein the size argument comprises a location of a second pointer associated with the buffer.

The applied art does not teach or suggest the above-cited language of independent claim 7. In particular, the applied art does not teach or suggest an annotation that comprises a property that takes a *size argument* and indicates a characteristic of the buffer, *where the size argument comprises a location of a second pointer associated with the buffer.*

Meyer does not teach or suggest the above-cited language of independent claim 7. Meyer describes "assertions" such as preconditions and postconditions for functions. [*See* Meyer at pp. 88-89.] However, Meyer's descriptions of assertions do not teach or suggest an annotation on a pointer to a buffer, where the annotation comprises a property that takes a size argument and indicates a characteristic of the buffer, and where the size argument comprises a location of a second pointer associated with the buffer.

The Splint manual also does not teach or suggest the above-cited language of independent claim 7. In particular, the Splint manual does not teach or suggest an annotation that comprises a property that takes a *size argument* and indicates a characteristic of the buffer, *where the size argument comprises a location of a second pointer associated with the buffer.* The Splint manual describes "maxRead" and "maxSet" notations. [*See* Splint manual at p. 48.] According to the Splint manual, "Given a buffer *b*, maxSet(b) denotes the highest address beyond *b* that can be safely used as an lvalue. . . . Similarly, maxRead denotes the highest index of a buffer that can be safely used [as] an rvalue." [*See id.*] maxSet and maxRead can take a reference to a buffer as an argument, but a reference to a buffer *is not a size argument.* Furthermore, the Splint manual's descriptions of "maxRead" and "maxSet" do not teach or suggest a size argument that comprises a location of a *second pointer associated with the buffer*.

For example, at page 14 the Application states, "The size argument of writableTo and readableTo can have several forms (i.e., size specifications)." Possible forms for the size argument include pointer locations such as "endpointer(location)" and "internalpointer(location)." [*See* Application at Table 7B.]

Claim 7 is allowable. Claims 14, 15 and 21 depend from claim 7 and are allowable for at least the reasons given above in support of claim 7. Therefore, the rejections of claims 7, 14, 15 and 21 under 35 U.S.C. § 102 should be withdrawn. Such action is respectfully requested.

*Independent Claim 24*

As amended, independent claim 24 recites in part:

> the annotation comprises a first instance of a keyword, the first instance of the keyword indicating that the first value satisfies all usability properties necessary to allow a first function to rely on the first value, wherein other instances of the keyword identical to the first instance are operable to indicate that other values having different respective value types satisfy all usability properties necessary to allow functions to rely on the respective other values, and wherein the usability properties depend on the value type.

The Splint manual does not teach or suggest the above-cited language of independent claim 24. In particular, the applied art does not teach or suggest an annotation that comprises a first instance of a keyword that indicates the first value satisfies all usability properties necessary to allow a first function to rely on the first value, *where other instances of the keyword identical to the first instance are operable to indicate that other values having different respective value types satisfy all usability properties necessary to allow functions to rely on the respective other values.*

The Splint manual describes several different annotations that can be used to provide information about various types. A simple example is the annotation "notnull" that can be used to annotate a pointer that is not null. [*See* Splint manual at p. 105 and Section 2.] Even if, for the sake of argument, an annotation such as "notnull" could be considered to indicate that a value of *a single type* (i.e., a pointer) satisfies all usability properties necessary to allow a function to rely on that value in some circumstances (i.e., when dereferencing the pointer), the annotations described in the Splint manual do not teach or suggest a keyword that also can be used to indicate that other values having *different value types* satisfy all usability properties necessary to allow functions to rely on the other values. Such a keyword can be useful, for example, to help reduce complexity of an annotation language by keeping the number of annotations in use small.

For example, at Table 9 the Application explains that the keyword "valid" can be used to annotate "any value" and "[s]tates that the value satisfies all usability properties of its type . . . .

For example, for a string buffer, *valid* means that the buffer pointer is not null, and the buffer is null-terminated." The keyword "valid" can be applied to types such as string buffers, pointers, structs, and others. "The usability of a value depends on the type of the value (e.g., its declared C type)." [*See* Application at p. 18.] In contrast, annotations such as "notnull" as described in the Splint manual might describe usability properties for a single type in some circumstances, but such annotations cannot also be used to *indicate that other values having different value types satisfy all usability properties necessary to allow functions to rely on the other values.*

Claim 24 is allowable. Claims 25-29 depend from claim 24 and are allowable for at least the reasons given above in support of claim 24. Therefore, the rejection of claims 24-29 under 35 U.S.C. § 102 should be withdrawn. Such action is respectfully requested.

*Independent Claim 30*

As amended, independent claim 30 recites in part:

> inserting an annotation having an argument in the computer program code, wherein the annotation annotates a value having a first declared value type with a first set of usability properties;
>
> wherein the annotation overrides the first set of usability properties of the first declared value type and indicates that the value has usability properties that depend on the properties of a second value type denoted by the argument of the annotation.

The Splint manual does not teach or suggest the above-cited language of independent claim 30. Regarding claim 30, the Examiner states, "As per claim 30, [Splint] discloses inserting an annotation . . . wherein the annotation indicates that the value has usability properties that depend on the properties of a second value type denoted by the argument of the annotation." [*See* Action at p. 10.] The Examiner cites p. 57 of the Splint manual and, specifically, the "alt" notation. [*See id.*]

Applicants respectfully disagree that the Splint manual teaches or suggest the elements of claim 30. At page 57, the Splint manual does describe use of the notation "/*@alt <type>, +@*/ . . . to indicate that an alternate type may be used." The "alt" notation described in the Splint manual indicates a possible alternate type, but the Splint manual does not teach or suggest the recited language of claim 30. In particular, the "alt" notation described in the Splint manual does not teach or suggest an annotation that *overrides a first set of usability properties of a first* *declared value type* and indicates that the value has usability properties that depend on the properties of a second value type denoted by the argument of the annotation.

For example, at page 20 the Application describes the "typefix" property and states, "the typefix property can be used to override the declared C/C++ type. The interpretation of valid for the annotated value is obtained from the type given by the typefix instead of the declared C/C++ type." In contrast, the "alt" notation described in the Splint manual only indicates a possible alternate type.

Claim 30 is allowable. Claims 31-33 depend from claim 30 and are allowable for at least the reasons given above in support of claim 30. Therefore, the rejection of claims 30-33 under 35 U.S.C. § 102 should be withdrawn. Such action is respectfully requested.

### *Independent Claim 34*

Independent claim 34 recites in part:

> including a size parameter with the annotation, wherein the size parameter describes a portion of the buffer to which the characteristic applies, and wherein the size parameter is operable to describe the portion of the buffer using a size specification selected from a group of plural different size specifications.

The Splint manual does not teach or suggest the above-cited language of independent claim 34. Regarding claim 34, the Examiner cites Section 9 of the Splint manual and, specifically, the "maxRead" and "maxSet" notations. [*See* Action at pp. 10-11.] The Examiner states, "[Splint] discloses . . . the size parameter is operable to describe the portion of the buffer using a size specification selected from a group of plural different size specifications . . . ." [*See id.*]

Applicants respectfully disagree that the Splint manual teaches or suggest the recited elements of claim 34. According to the Splint manual, "Given a buffer *b*, maxSet(b) denotes the highest address beyond *b* that can be safely used as an lvalue. . . . Similarly, maxRead denotes the highest index of a buffer that can be safely used [as] an rvalue." [*See* Splint manual at p. 48.] Even if, for the sake of argument, "maxSet" and "maxRead" could be considered size parameters, the Splint manual's descriptions of "maxRead" and "maxSet" do not teach or suggest a size parameter that is operable to describe a portion of a buffer *using a size specification selected from a group of plural different size specifications.*

For example, at page 14 the Application states, "The size argument of writableTo and readableTo can have several forms (i.e., size specifications)." Possible size specifications include byte counts, element counts, sentinel position, and pointer locations such as "endpointer(location)" and "internalpointer(location)." [*See* Application at Table 7B.] In contrast, the "maxSet" and "maxRead" notations described in the Splint manual are described as only denoting a buffer index/address.

Claim 34 is allowable. Claims 35-37 depend from claim 34 and are allowable for at least the reasons given above in support of claim 34. Therefore, the rejection of claims 34-37 under 35 U.S.C. § 102 should be withdrawn. Such action is respectfully requested.

### *New Claims*

Claims 41-55 have been added. Claims 41-45 depend directly or indirectly from claim 7 and are allowable for at least the reasons given above in support of their respective parent claims. Claim 46 depends from claim 24 and is allowable for at least the reasons given above in support of claim 24.

Independent claim 47 is allowable because the applied art does not teach or suggest, for example, "the at least one code annotation comprises an explicit dereference qualifier, and wherein the explicit dereference qualifier is operable to specify one or more properties *of each of the plural elements of the data structure*." New claims 48 and 49 depend from claim 47 and are allowable for at least the reasons given above in support of claim 47.

Independent claim 50 includes language similar to language in claim 7 and is allowable because the applied art does not teach or suggest, for example, "the at least one annotation comprises a property that indicates a characteristic of the buffer, wherein the property that indicates the characteristic of the buffer takes a size argument, and *wherein the size argument comprises a location of a second pointer associated with the buffer*." New claims 51 and 52 depend from claim 50 and are allowable for at least the reasons given above in support of claim 50.

Independent claim 53 includes language similar to language in claim 30 and is allowable because the applied art does not teach or suggest, for example, "the annotation *overrides the first set of usability properties of the first declared value type* and indicates a second set of usability properties for the value that depend on the second value type denoted by the argument of the annotation" New claims 54 and 55 depend from claim 53 and are allowable for at least the reasons given above in support of claim 53.

Support for new claims 41-45 can be found, for example, in the Application at Table 7B. Support for new claim 46 can be found, for example, in the Application at Table 3. Support for new claims 47-49 can be found, for example, in the Application at Table 2. Support for new claims 50-52 can be found, for example, in the Application at Table 7B. Support for new claims 53-55 can be found, for example, in the Application at Table 11.

## *Request for Interview*

If any issues remain, the Examiner is formally requested to contact the undersigned attorney prior to issuance of the next Office Action in order to arrange a telephonic interview. It is believed that a brief discussion of the merits of the present application may expedite prosecution. Applicants submit the foregoing formal Amendment so that the Examiner may fully evaluate Applicants' position, thereby enabling the interview to be more focused.

This request is being submitted under MPEP § 713.01, which indicates that an interview may be arranged in advance by a written request.

## *Conclusion*

The claims in their present form should now be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By [illegible]
Stephen A. Wight
Registration No. 37,759